AOU
الجامعة العربية المفتوحة
Arab Open University
Egypt

91 Management Information System

*STAGES OF DBS DEVELOPMENT LIFECYCLE

*ERD

*ERD MAPPING

AOU UNIVERSITY

Database Planning

- During planning phase, four major activities are performed:
  - Review and approve the database project request.
  - Prioritize the database project request.
  - Allocate resources such as money, people and tools.
  - Arrange a development team to develop the database project.

Database Planning

- **Two steps:**

1. **Mission Statement** : Major aims for the Database Project.

2. **Mission Objectives** : Tasks that DB should support, *Driven through Fact Finding techniques: interviews, questionnaire.*

Database Planning
Mission Statement

- **Interviewing the business director:**
  - What is the purpose of your company?
  - Why do you need the DB?
  - How can the DB solve your pr

Database Planning
Mission Objectives

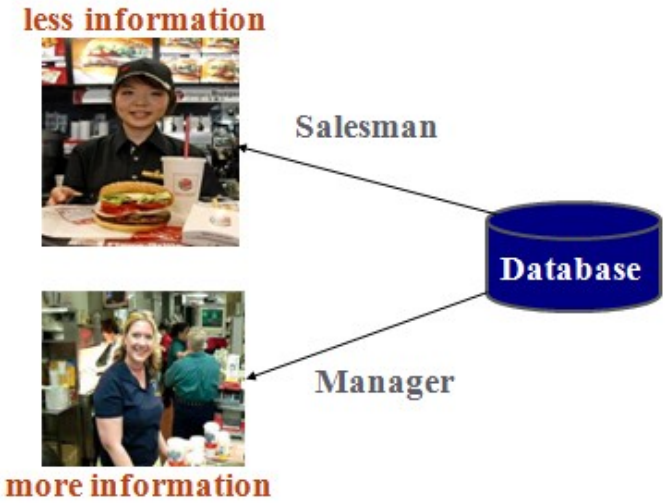
- **Interviewing different staff:**
  - Job description.
  - Daily tasks.
  - Data that they deal with.
  - Daily reports.
  - Type of things they keep track on.
  - Customer services.

Database Planning
Mission Objectives

1. To manage (add, delete, update) data on branches.
2. To manage (add, delete, update) data on staff.
3. To manage (add, delete, update) on meals.
4. To manage (add, delete, update) on clients.
5. To perform searches on branches.
6. To perform searches on staff.
7. To perform searches on meals.
8. To perform searches on clients

System Definition

- Identify User views; Define what is required of a database application from the perspective of:
  - A particular job role (such as Manager or Supervisor) or
  - Enterprise application area (such as marketing or personnel).



Example: *User Views*

Requirements Collection & Analysis

- Identifying the required functionality for a database system is crucial, as systems with inadequate functionality will fail.

- Information is gathered for each major user view including:
  - a description of data used or generated;
  - details of how data is to be used/generated;
  - any additional requirements for new

Database Design

- **Main aims:**
  - To represent data and relationships required
    by users and applications.
  - To provide a data model which supports transactions.
  - To specify a design that meets performance requirements.

Database Design

- **Three main phases of DB design:**
  - Conceptual Database Design.
    - Create a conceptual data model Independent of any implementation details.
  - Logical Database Design. (What)
    - At this point you know which type of DBMS you will implementing in - e.g. relational, object-oriented etc (but not the actual DBMS).
  - Physical Database Design. (How)
    - Derive tables & constraints.
    - Identify storage structures and access methods.
    - Design security features.

Implementation

- Physical realization of the database and application designs.
  - Create database schemas and empty database files. (DDL)
  - Create the application programs. (DML and JAVA or C++ ...)

Testing

- Process of running the database system with intent of finding errors.

- Demonstrates that database and application programs appear to be working according to requirements.

Operational Maintenance

- Process of monitoring and maintaining database system following installation.

- **Operational Maintenance Activities:**

  - Monitoring performance of system.
  - Maintaining and upgrading database applica
  - Incorporating new requirements into DB app

Database Design

- Before we look at how to create and use a database we'll look at how to design one

- Need to consider
  - What tables, keys, and constraints are needed?
  - What is the database going to be used for?

- Conceptual design
  - Build a model independent of the choice of DBMS

- Logical design
  - Create the database in a given DBMS

- Physical design
  - How the database is stored in hardware

ENTITY/RELATIONSHIP MODELLING

- **E/R Modelling is used for conceptual design**
  - Entities
  - Attributes
  - Relationships
- Example
  - In a University database we might have entities for Students, Modules and Lecturers. Students might have attributes such as their ID, Name, and Course, and could have relationships with Modules and Lecturers

DATA MODELING USING THE ENTITY-RELATIONSHIP (ER) MODEL

- **Entity-Relationship (ER) model**
  - Popular high-level conceptual data model
- **ER diagrams**
  - Diagrammatic notation associated with the ER model

EXAMPLE COMPANY DATABASE

Requirements for the COMPANY Database:

- The company is organized into DEPARTMENTs, department has a name, number, and an employee who *manages* the department.

We keep track of the start date of the department manager. A department may have several locations.

- Each department *controls* a number of PROJECTs. Each project has a name, number, and is located at a single location.

- We store each EMPLOYEE's social security number, address, salary, sex, and birth date. Each employee *works for* one department but may *work on* several projects.

- We keep track of the number of hours per week that an employee currently works on each project.
- We also keep track of the *direct supervisor* of each employee
- Each employee may have a number of DEPENDENTs.
- For each dependent, we keep their name, sex, birth date, and relationship to the employee.

ER MODEL CONCEPTS (POPULAR CONCEPTUAL MODEL)

3.1 ENTITIES AND ATTRIBUTES

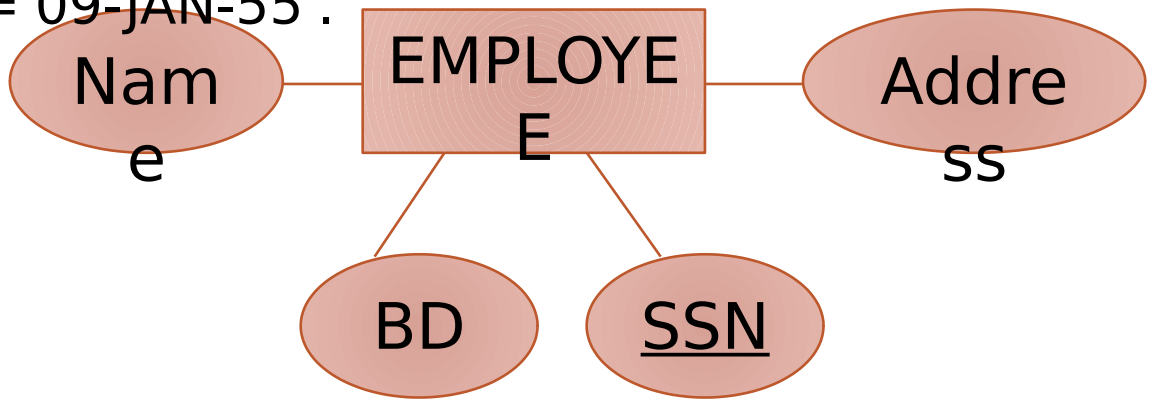
- **Entities** are specific objects or *things* in the mini-world that are represented in the database; for example the EMPLOYEE John Smith, the Research DEPARTMENT, the ProductX PROJECT.

Entity EMPLOYEE

- **Attributes** are *properties* used to describe an entity; for example an EMPLOYEE entity may have a Name, SSN, Address, Sex, BirthDate.

Attribute

- A *specific entity* will have a **value** for each of its attributes; for example a specific employee entity may have Name='John Smith', SSN='123456789', Address='731 Fondren, Houston, TX', Sex='M', BirthDate='09-JAN-55'.

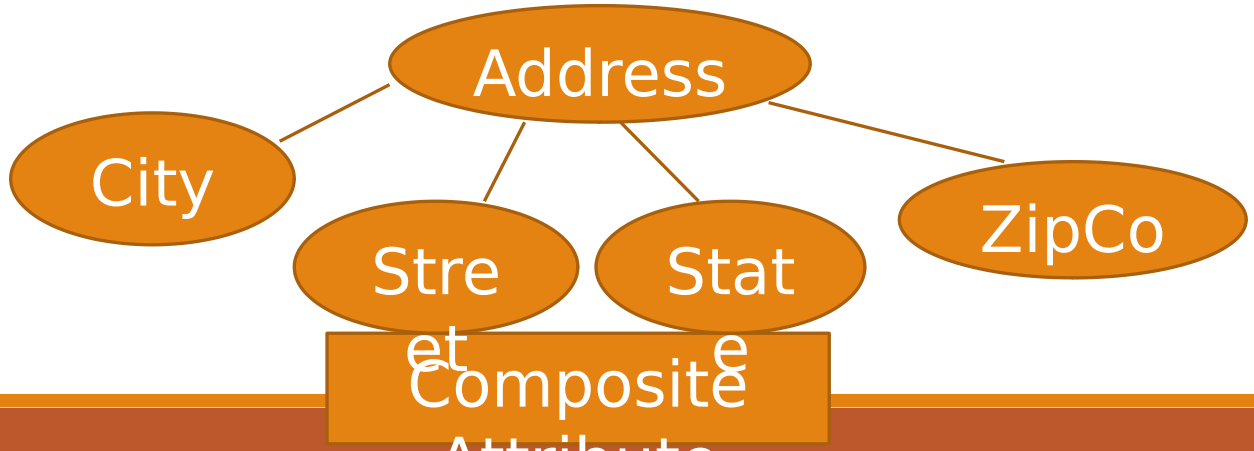


TYPES OF ATTRIBUTES

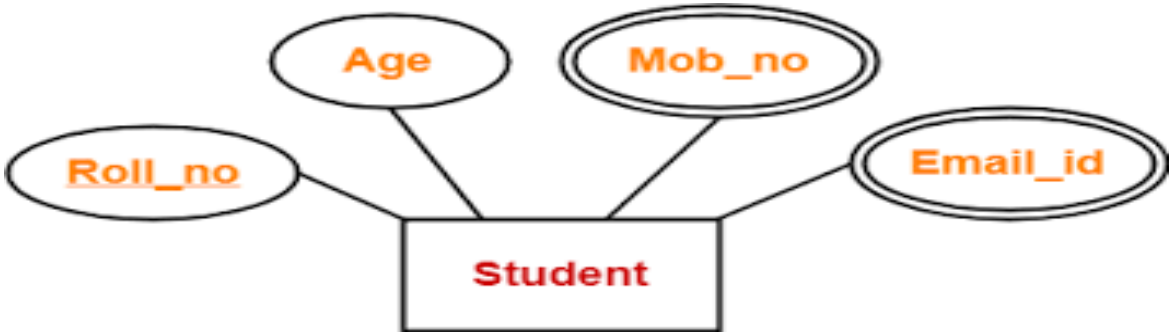
- **Simple**: Each entity has a *single atomic value* for the attribute; for example SSN or Sex.

- **Composite**: The attribute may be composed of several components; for example Address(Apt#, House#, Street, City, State, ZipCode, Country) or Name(FirstName, MiddleName, LastName).

- Composition may form a hierarchy where some components are themselves composite.



Multi-valued: An entity may have *multiple values* for that attribute; for example Color of a CAR or PreviousDegrees of a STUDENT or Locations of a DEPARTMENT. Denoted as {Color} or {PreviousDegrees} or {Locations}.



Multivalued Attribute

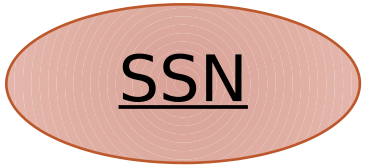
3.2 ENTITY TYPES AND KEY ATTRIBUTES

Entities with the same basic attributes are grouped or *typed* into an **entity type**.

- For example, the EMPLOYEE entity type or the PROJECT entity type.

An attribute of an entity type for which each entity must have a *unique value* is called a **key attribute** of the entity type.

For example SSN of EMPLOYEE.



Key Attribute

- A key attribute may be composite. For example, vehicle Registration Number is a key of the CAR entity type with components (Number, State).

- An entity type may have *more than one* key. For example, the CAR entity type may have two keys:

  VehicleIdentificationNumber and

  VehicleRegistrationNumber(Number, State).

FIGURE 3.7 THE CAR ENTITY TYPE, WITH TWO KEY ATTRIBUTES REGISTRATION AND VEHICLEID. MULTIVALUED ATTRIBUTES ARE SHOWN BETWEEN SET BRACES {}. COMPONENTS OF A COMPOSITE ATTRIBUTE ARE SHOWN BETWEEN PARENTHESES ().

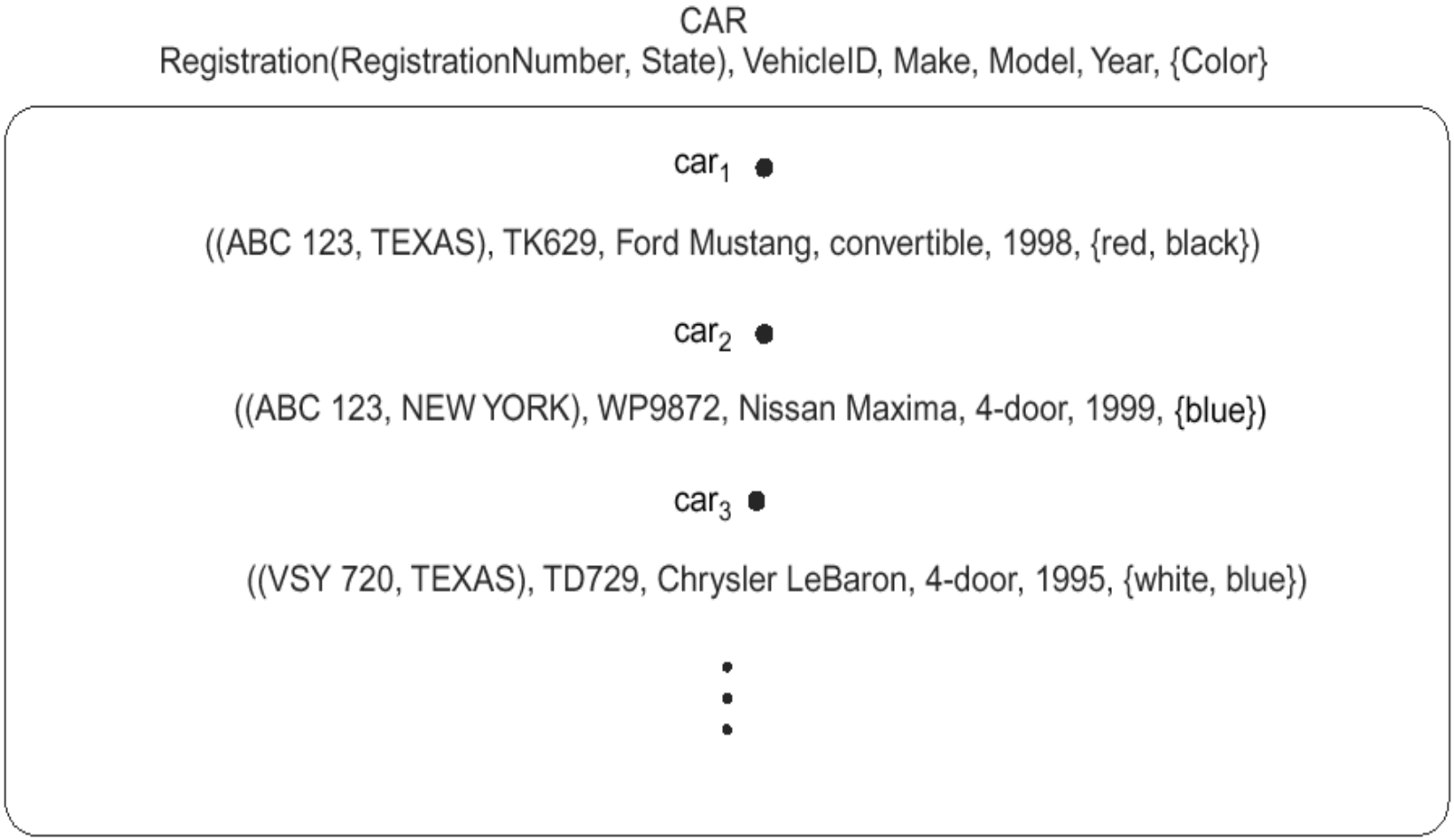

CAR
Registration(RegistrationNumber, State), VehicleID, Make, Model, Year, {Color}
car1
((ABC 123, TEXAS), TK629, Ford Mustang, convertible, 1998, {red, black})
car2
((ABC 123, NEW YORK), WP9872, Nissan Maxima, 4-door, 1999, {blue})
car3
((VSY 720, TEXAS), TD729, Chrysler LeBaron, 4-door, 1995, {white, blue})

FIGURE 3.8 PRELIMINARY DESIGN OF ENTITY TYPES FOR THE COMPANY DATABASE WHOSE REQUIREMENTS ARE DESCRIBED IN

DEPARTMENT

Name, Number, {Locations}, Manager, ManagerStartDate

PROJECT

Name, Number, Location, ControllingDepartment

EMPLOYEE

Name (FName, MInit, LName), SSN, Sex, Address, Salary, BirthDate, Department, Supervisor, {WorksOn (Project, Hours)}

DEPENDENT

Employee, DependentName, Sex, BirthDate, Relationship

3.3 RELATIONSHIPS AND RELATIONSHIP TYPES

- A **relationship** relates two or more distinct entities with a specific meaning

- For example, EMPLOYEE John Smith *works on* the ProductX PROJECT .

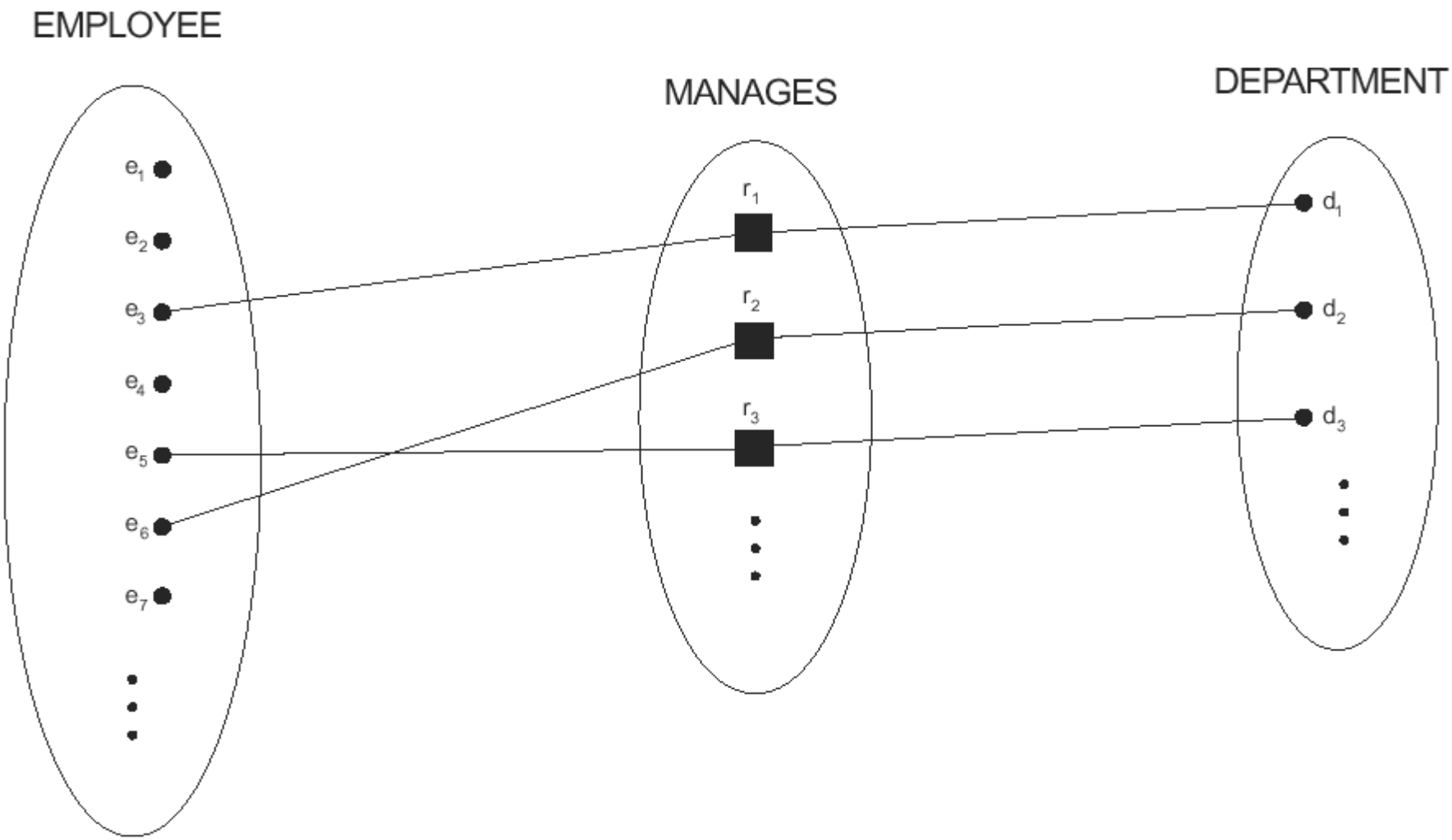
- Relationships of the same type are grouped or *typed* into a **relationship type**.

- For example, the WORKS_ON relationship type in which EMPLOYEEs and PROJECTs participate.

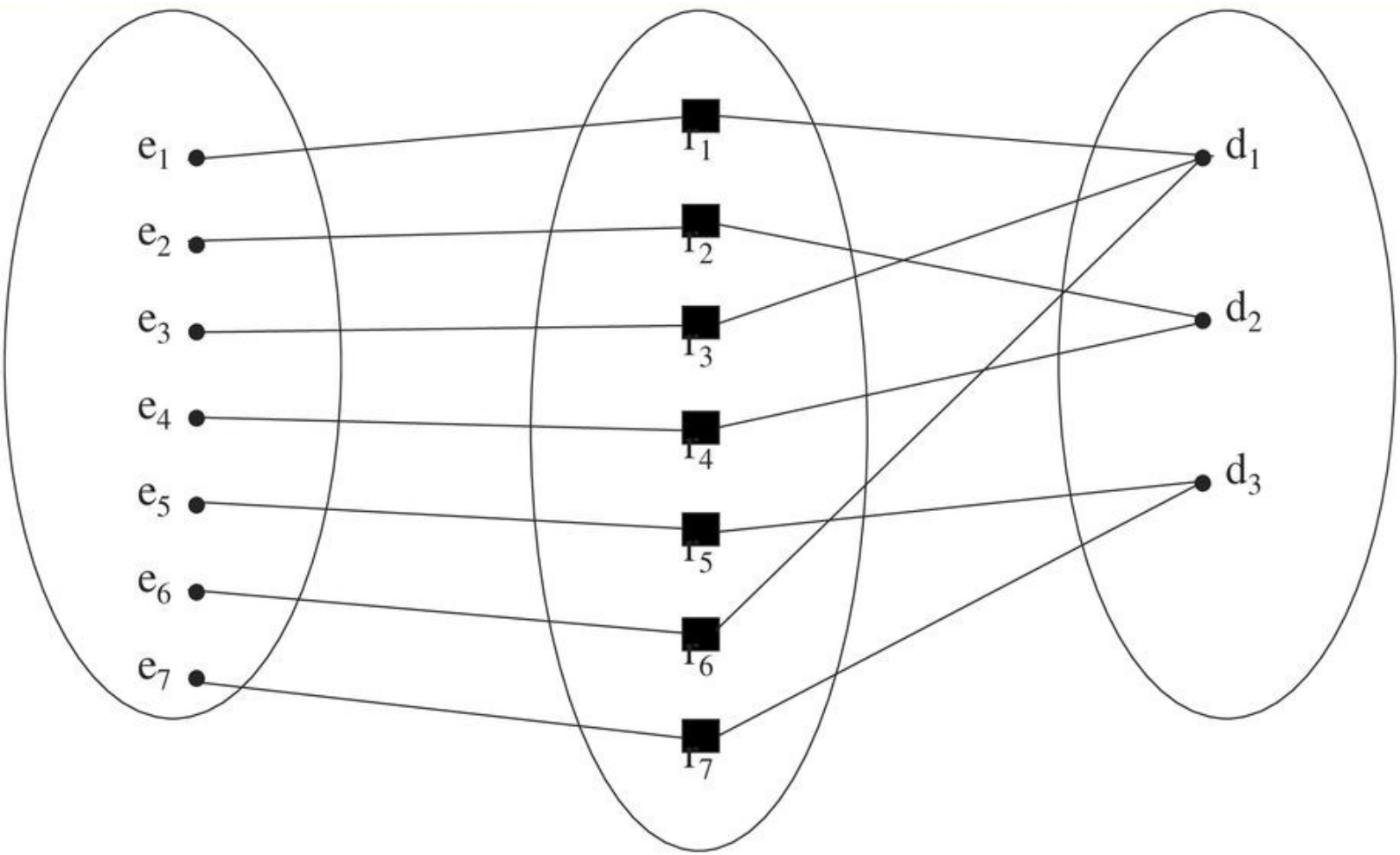
- The **degree** of a relationship type is the number of **participating** entity types. Both MANAGES and WORKS_ON are **binary relationships**.

Relationship Types

Figure 3.12 The 1:1 relationship MANAGES

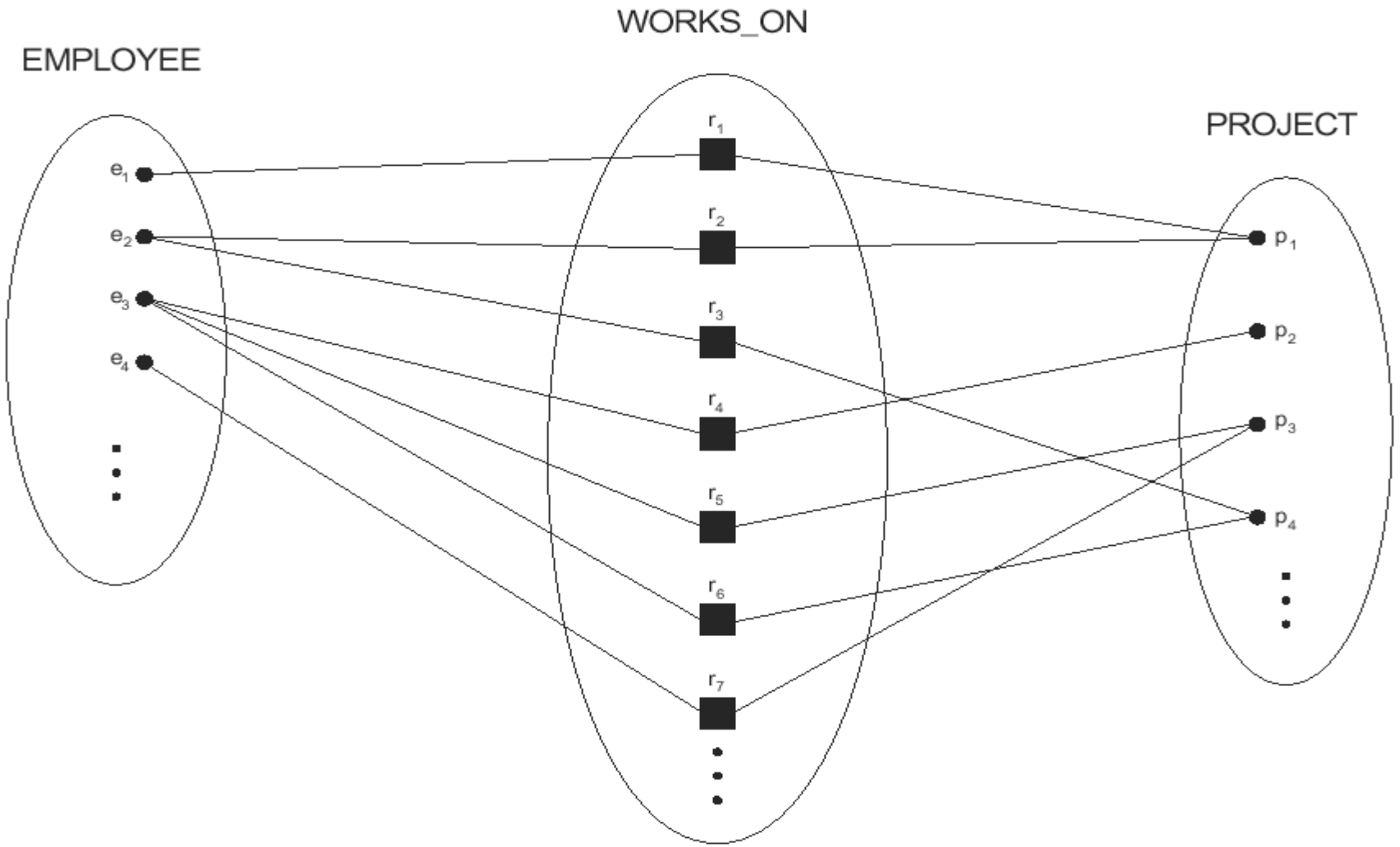


One to Many Relationship



Slide 3- 5

Figure 3.13 The M:N relationship WORKS_ON between EMPLOYEE and PROJECT.



Examples:

(a) A department has *exactly one* manager.

- Specify (1,1) for participation of Department MANAGES

An employee can manage *at most one* department.

- Specify (0,1) for participation of EMPLOYEE in MANAGES

(b) An employee can work for *exactly one* department.

- Specify (1,1) for participation of EMPLOYEE in WORKS_FOR

A department can have *any number of* employees.

-Specify (1,n) for participation of DEPARTMENT in WORKS_FOR

3.4 Weak Entity Types

- An entity type that does not have a key attribute

-A weak entity type must participate in an **identifying relationship type** with an **owner** or **identifying entity type**

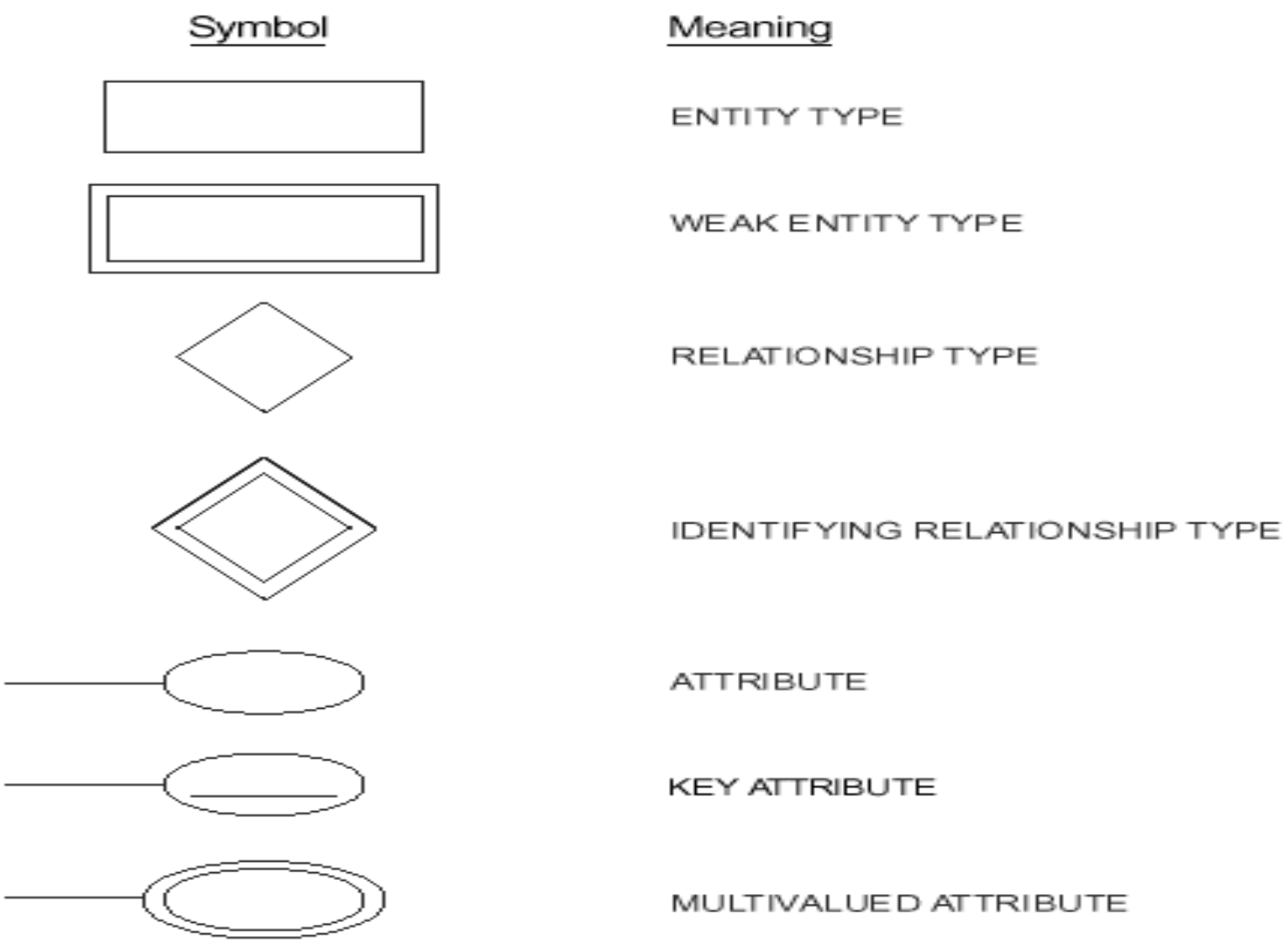
-

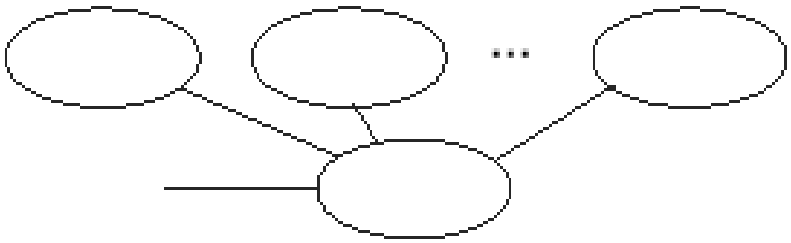
Example:

Suppose that a DEPENDENT entity is identified by the dependent's first name and birthdate, *and* the specific EMPLOYEE that the dependent is related to.

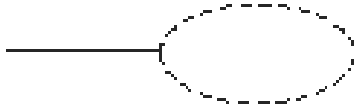
DEPENDENT is a weak entity type with EMPLOYEE as its identifying entity type via the identifying relationship type DEPENDENT_OF.

Figure 3.14 Summary of ER diagram notation

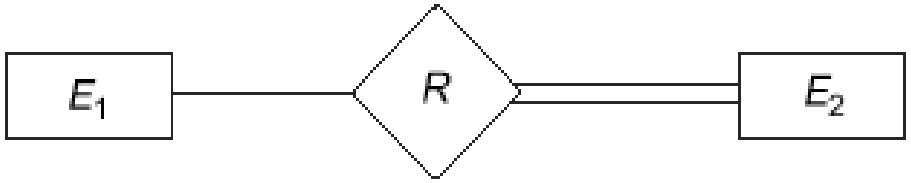




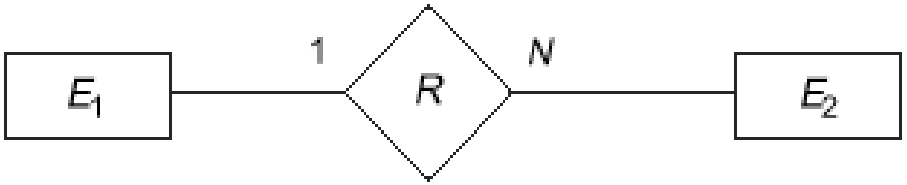
COMPOSITE ATTRIBUTE



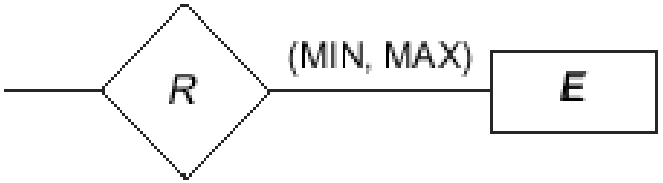
DERIVED ATTRIBUTE



TOTAL PARTICIPATION OF E_2 IN R

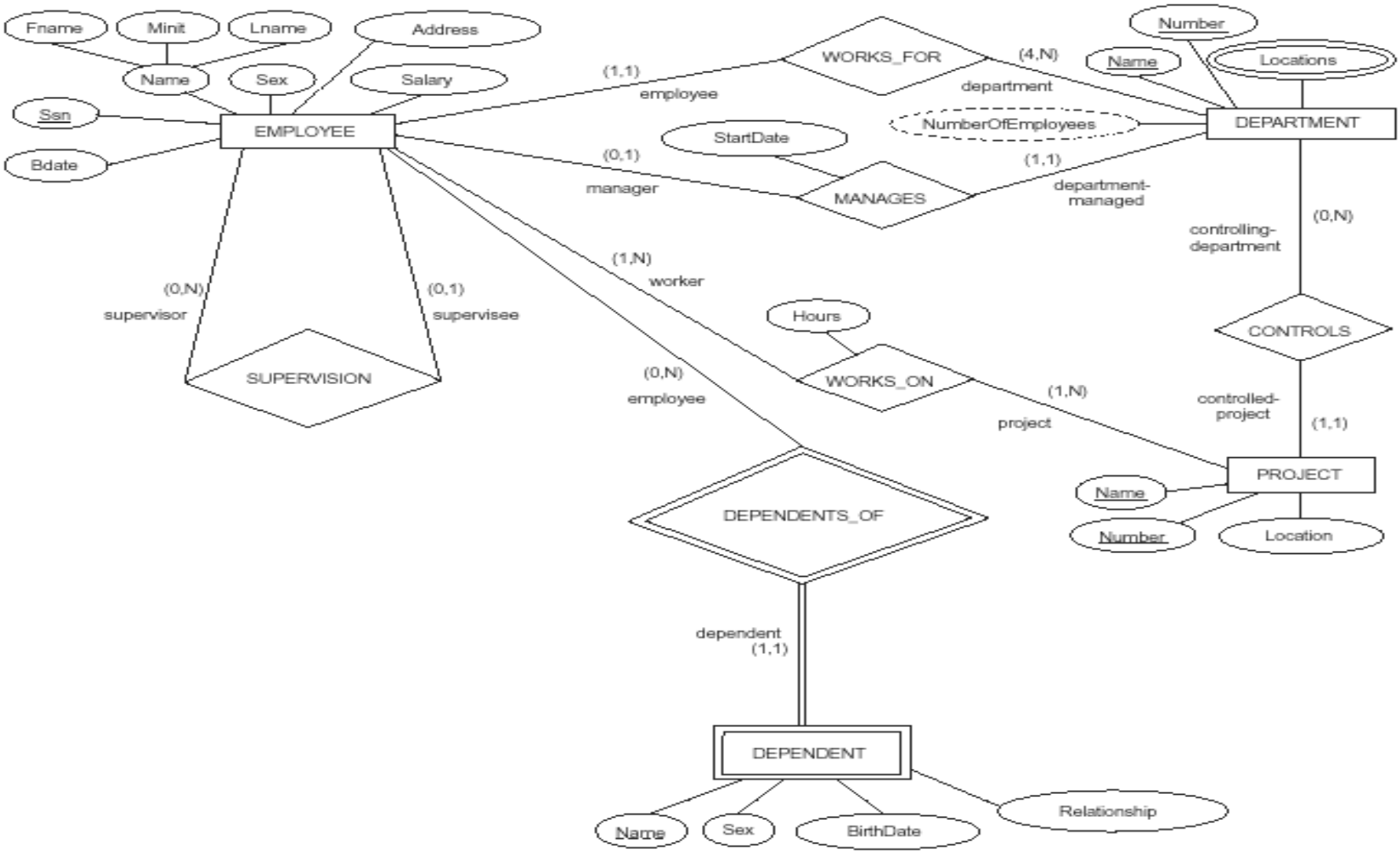


CARDINALITY RATIO 1: N FOR E_1:E_2 IN R



STRUCTURAL CONSTRAINT (min, max)
ON PARTICIPATION OF E IN R

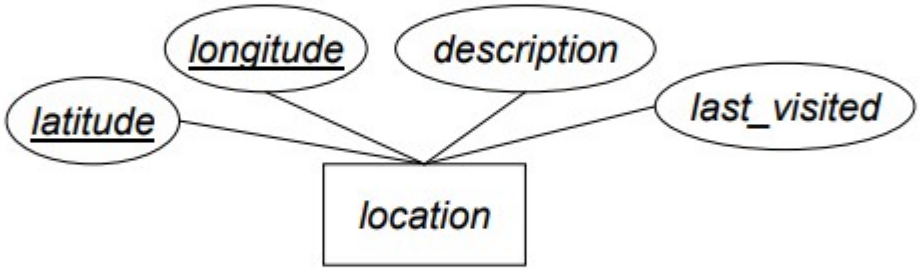
Figure 3.15 ER diagram for the COMPANY schema, with all role names included and with structural constraints on relationships specified using the alternate notation (min, max).



Mapping an ERD to Relational Model

Entity-Set Examples

- Geocache location E-R diagram:

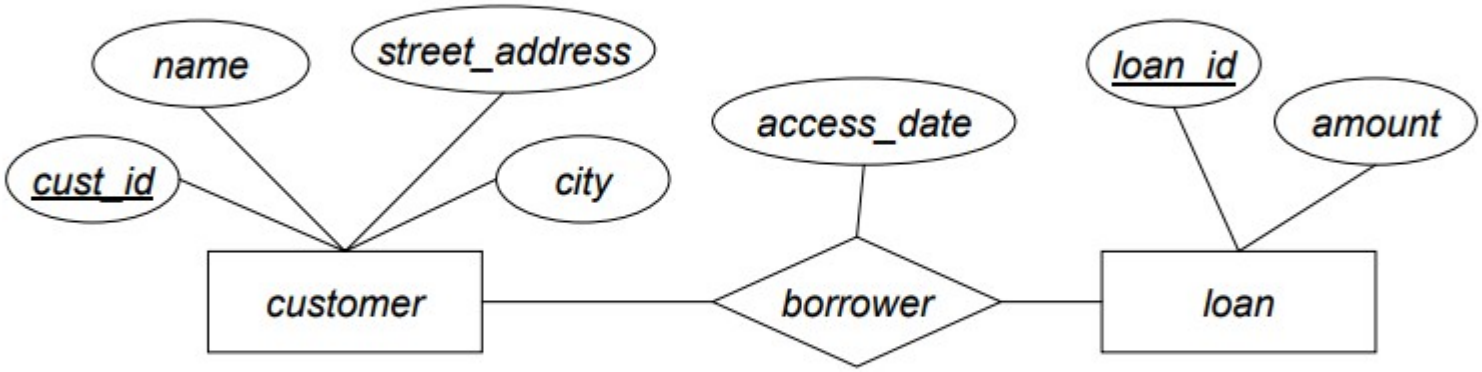


  - Entity-set named *location*
- Convert to relation schema:

  location(*latitude*, *longitude*, *description*, *last_visited*)

Entity-Set Examples (2)

- E-R diagram for customers and loans:



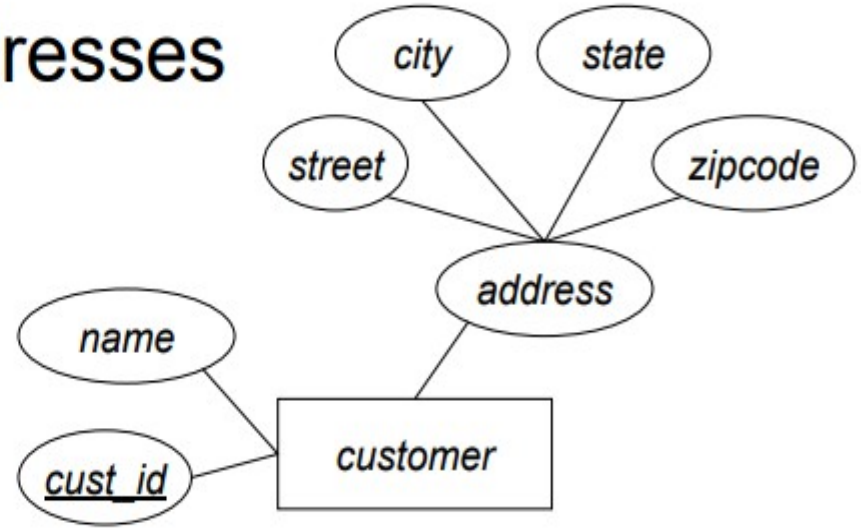
- Convert *customer* and *loan* entity-sets:

  customer(*cust_id*, *name*, *street_address*, *city*)

  loan(*loan_id*, *amount*)

Composite Attribute Example

- Customers with addresses

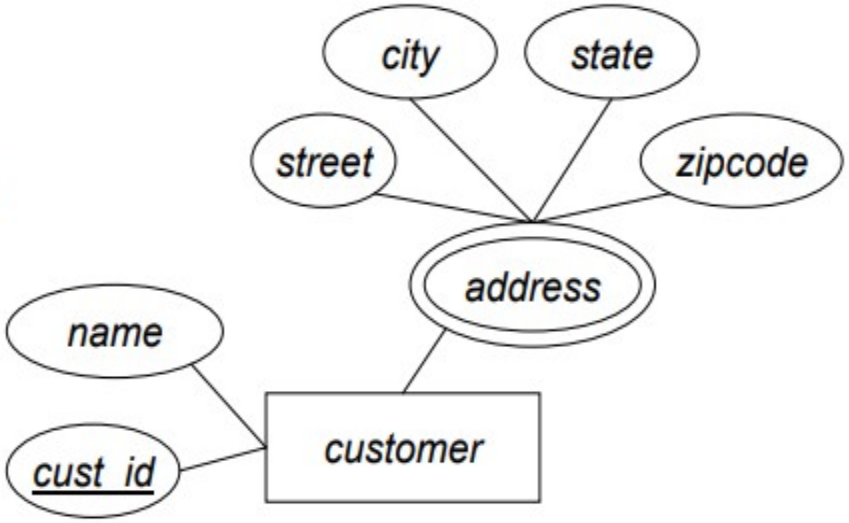


- Each component of *address* becomes a separate attribute

  customer(*cust_id*, *name*, *street*, *city*, *state*, *zipcode*)

Multivalued Attribute Example

- Customers with multiple addresses



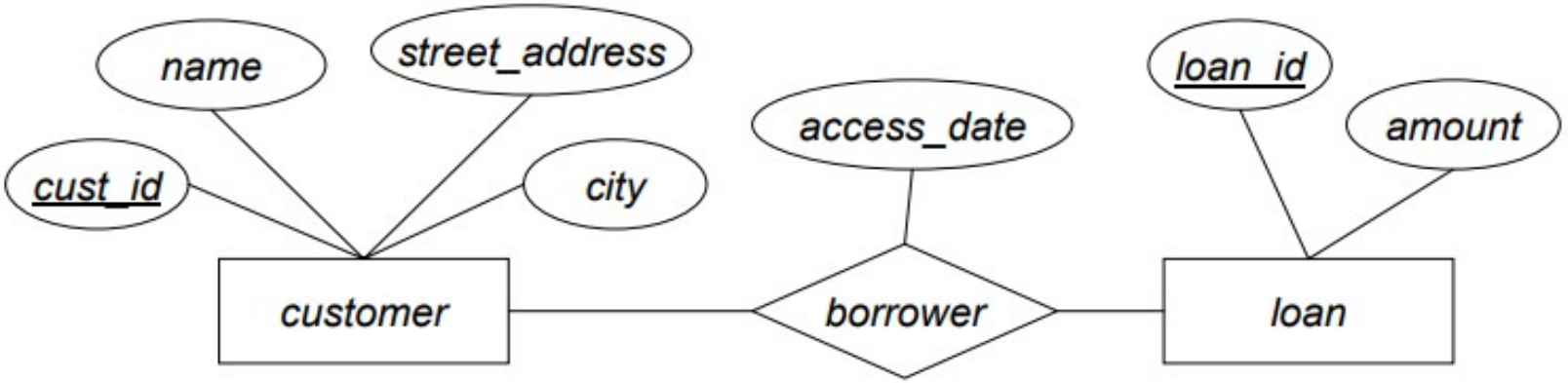
- Create separate relation to store each address

  customer(*cust_id*, *name*)

  cust_addrs(*cust_id*, *street*, *city*, *state*, *zipcode*)

  – Large primary keys aren't ideal – tend to be costly

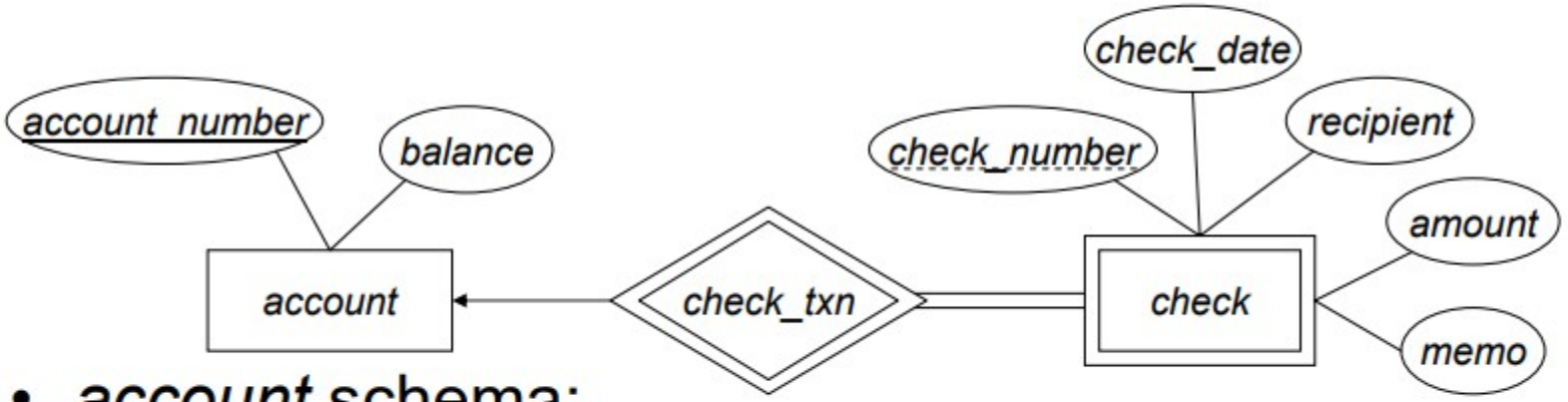
Relationship-Set Example



- Relation schema for *borrower*:
  - Primary key of *customer* is *cust_id*
  - Primary key of *loan* is *loan_id*
  - Descriptive attribute *access_date*
  - *borrower* mapping cardinality is many-to-many

    borrower(*cust_id*, *loan_id*, *access_date*)

Weak Entity-Set Example



- *account* schema:
  account(*account_number*, *balance*)
- *check* schema:
  - Discriminator is *check_number*
  - Primary key for *check* is:
    (*account_number*, *check_number*)

  check(*account_number*, *check_number*,
  check_date, *recipient*, *amount*, *memo*)